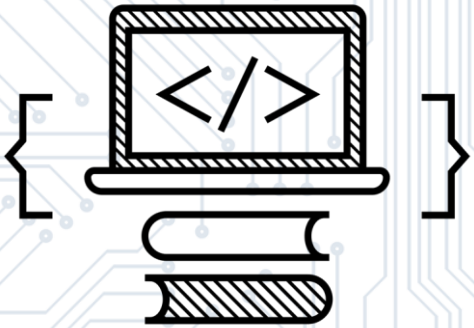
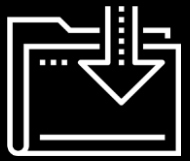


JavaScript Juggernauts

Web Development Boot Camp

Lesson 3.3



Today’s Class

Objectives

In today's class, we'll cover:

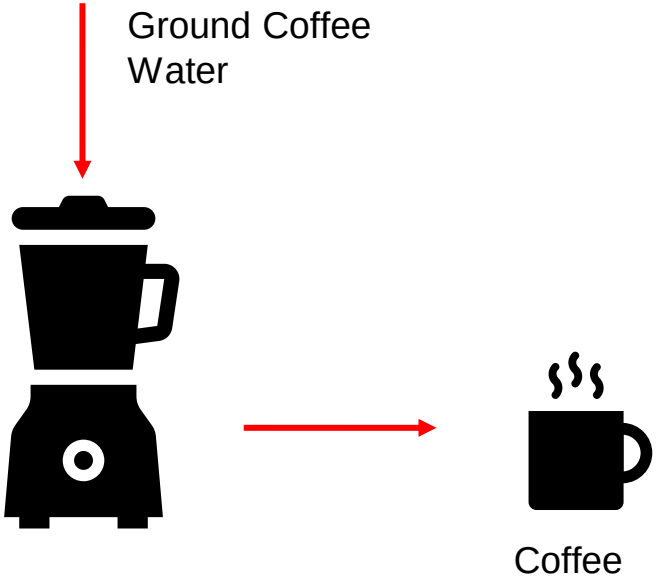
JavaScript Functions

JavaScript Objects

Building Simple JavaScript Applications

JavaScript Functions

Lets’ Visualize a Function



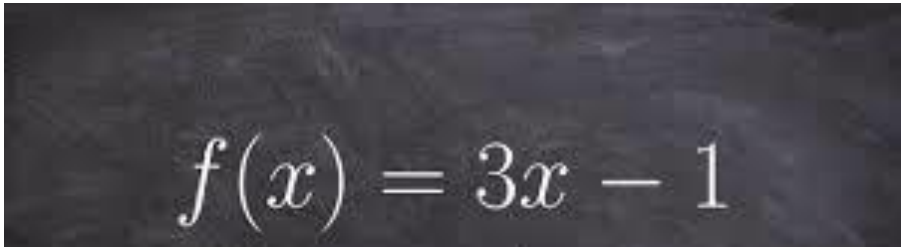
Let x = scoops of coffee
Let y = cups of water

Let z = cups of coffee

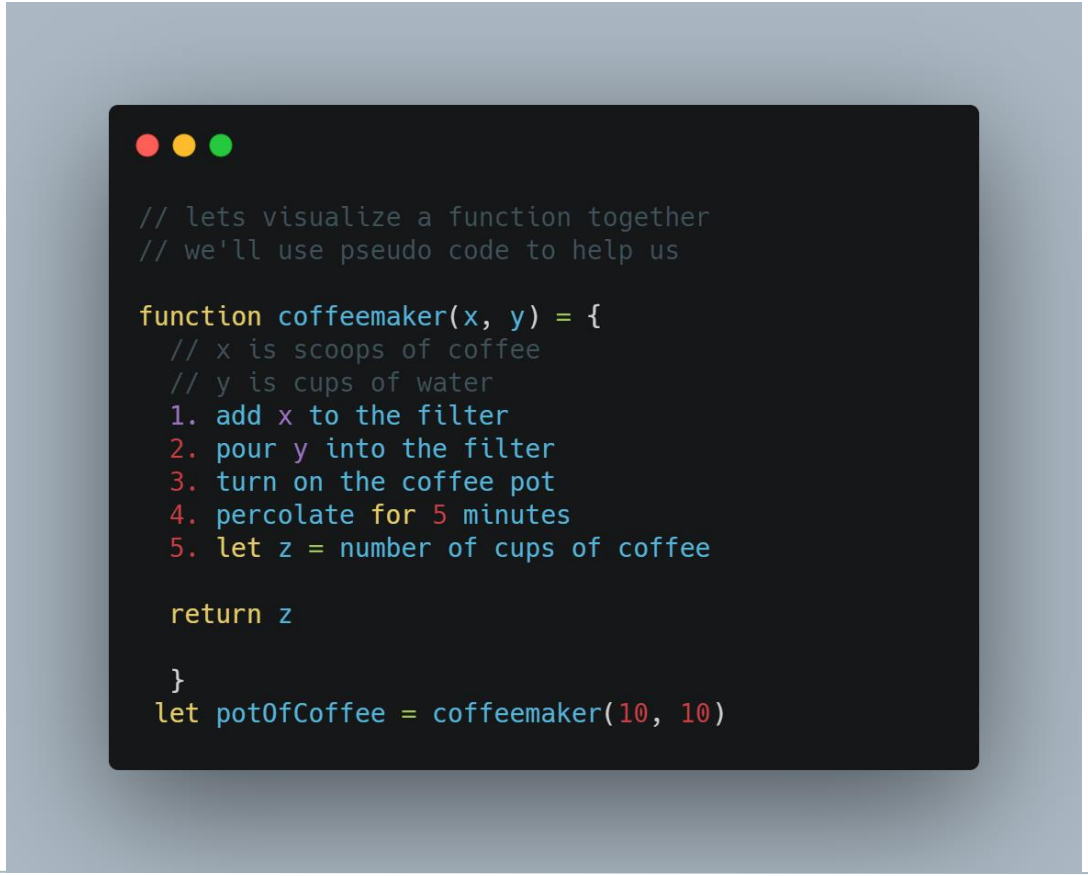
$$f(x, y) = z$$

Parameters
or
Arguments

Result


$f(x) = 3x - 1$

Visualizing our Coffee Making Function



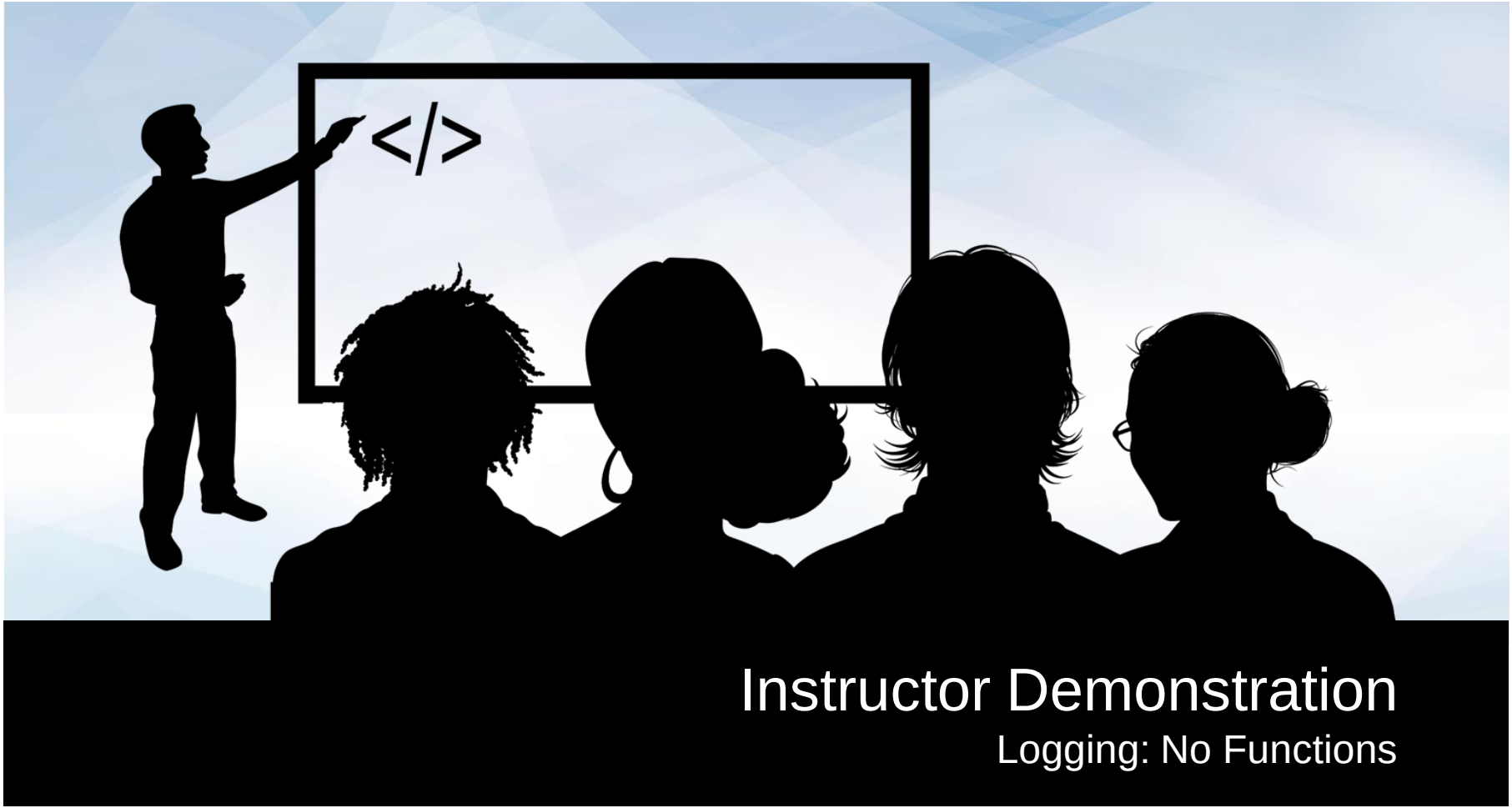
```
// lets visualize a function together
// we'll use pseudo code to help us

function coffeemaker(x, y) = {
  // x is scoops of coffee
  // y is cups of water
  1. add x to the filter
  2. pour y into the filter
  3. turn on the coffee pot
  4. percolate for 5 minutes
  5. let z = number of cups of coffee

  return z

  }
 let potOfCoffee = coffeemaker(10, 10)
```

$$f(x, y)$$



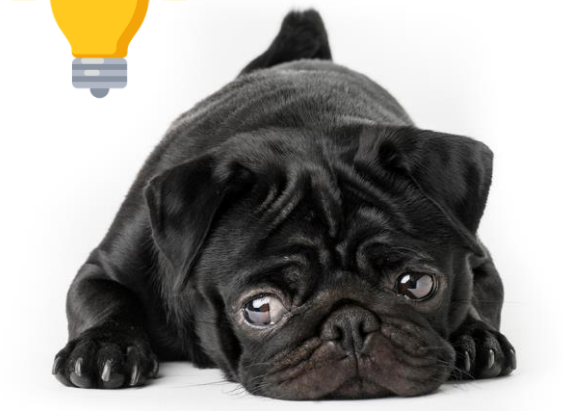
Instructor Demonstration

Logging: No Functions

Mondo Repetitive

Who wants to maintain this?

Hint: No one.



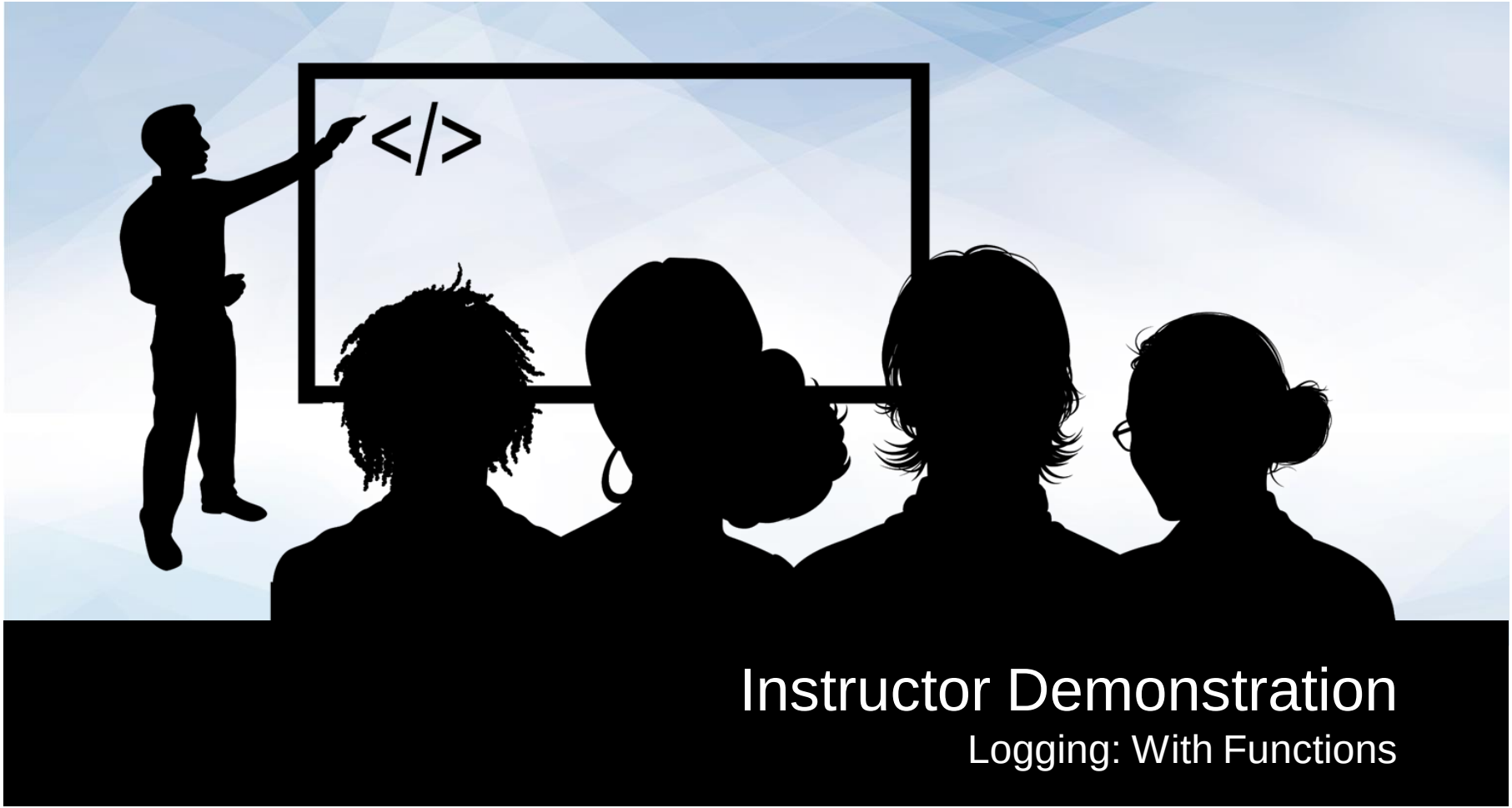
```
// For Loop for Brands
for (var i = 0; i < brands.length; i++) {
  console.log(brands[i]);
}
console.log("---------");

// For Loop for Heroes
for (var i = 0; i < heroes.length; i++) {
  console.log(heroes[i]);
}
console.log("---------");

// For Loop for booksOnMyShelf
for (var i = 0; i < booksOnMyShelf.length; i++) {
  console.log(booksOnMyShelf[i]);
}
console.log("---------");

// For Loop for thingsInFrontOfMe
for (var i = 0; i < thingsInFrontOfMe.length; i++) {
  console.log(thingsInFrontOfMe[i]);
}
console.log("---------");

// For Loop for howIFeel
for (var i = 0; i < howIFeel.length; i++) {
  console.log(howIFeel[i]);
}
console.log("---------");
```



Instructor Demonstration

Logging: With Functions

Much Better with Functions!

Squeaky clean code. Minimal repetition.

```
// Here we create a "Function" that allows us to "call" (run) the loop for any array we wish.
// We pass in an array as an "argument".
function consoleInside(arr) {

  // We then loop through the selected array.
  for (var i = 0; i < arr.length; i++) {

    // Each time we print the value inside the array.
    console.log(arr[i]);
  }
  console.log("---------");
}
```

Partner Activity:
My First Functions

Suggested Time:
20 minutes

Partner Activity: My First Functions

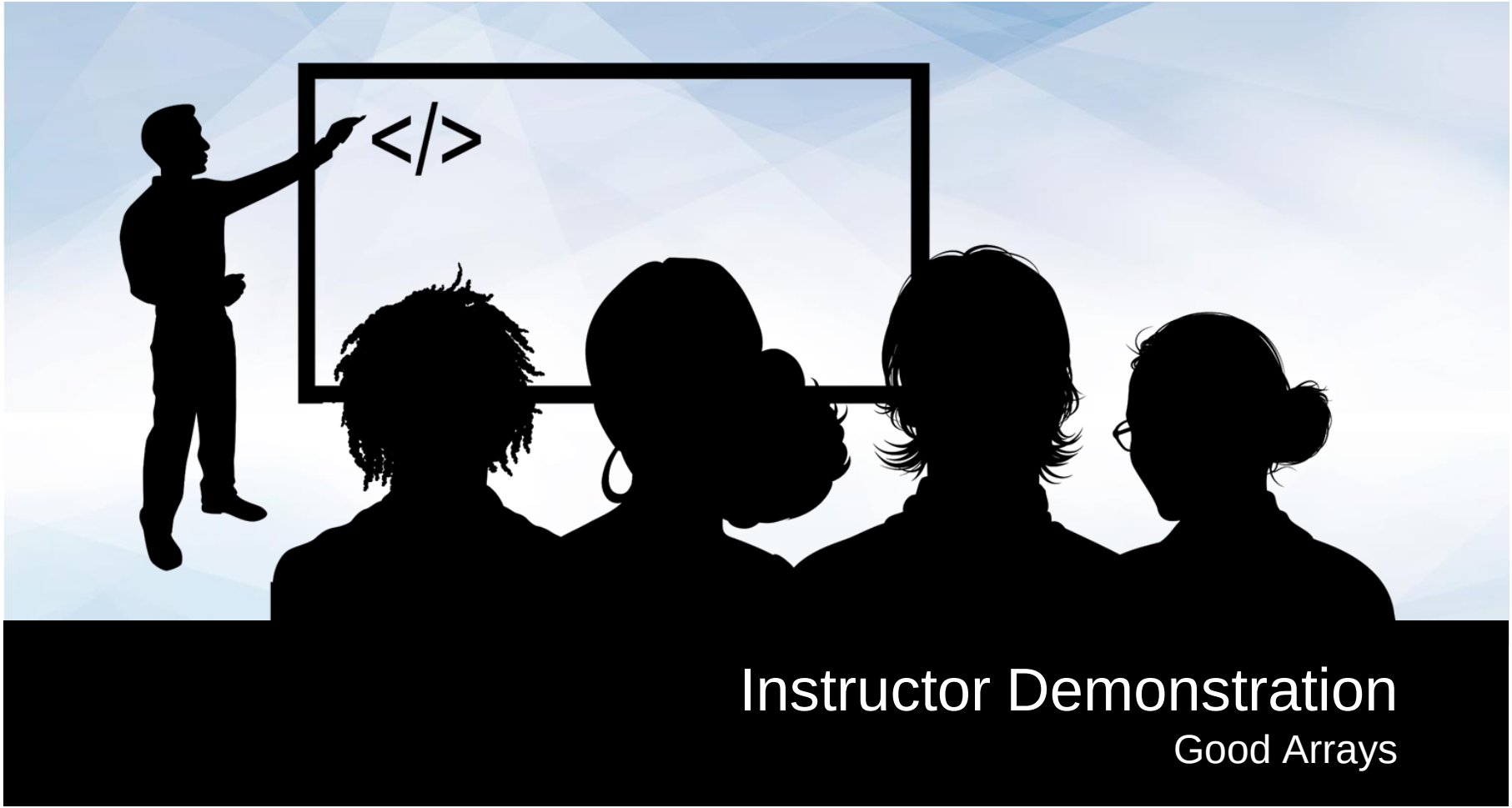
Working in pairs and using the starter file sent to you via Slack, fill in the missing functions and function calls.

Note: Try to finish all four functions if you can, but don't be distressed if you only get 1 or 2. The important thing is that you get at least one function fully done.

HINT: Look back to the previous example if you need help.

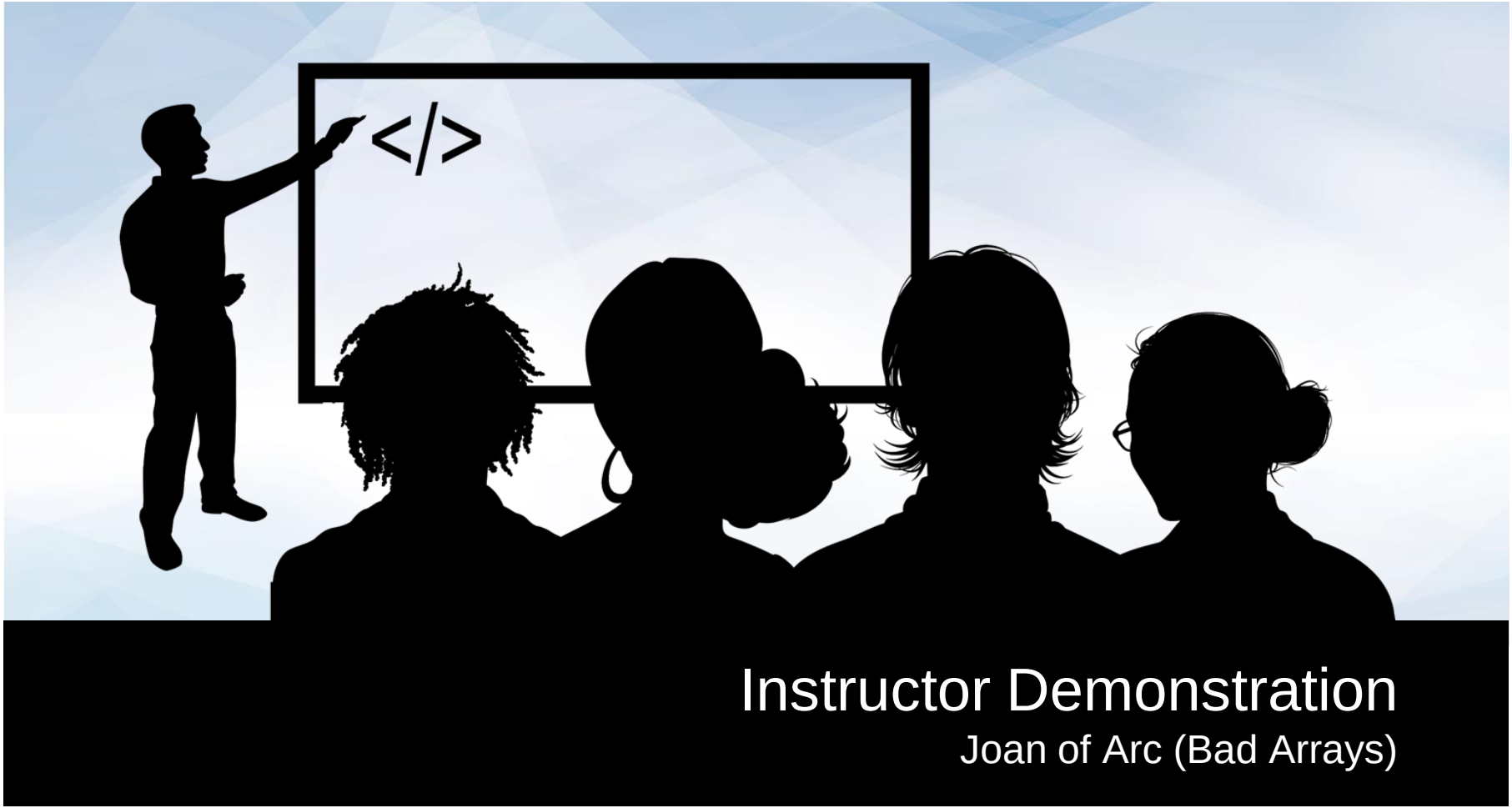
Suggested Time: 20 minutes

JavaScript Objects



Instructor Demonstration

Good Arrays



Instructor Demonstration

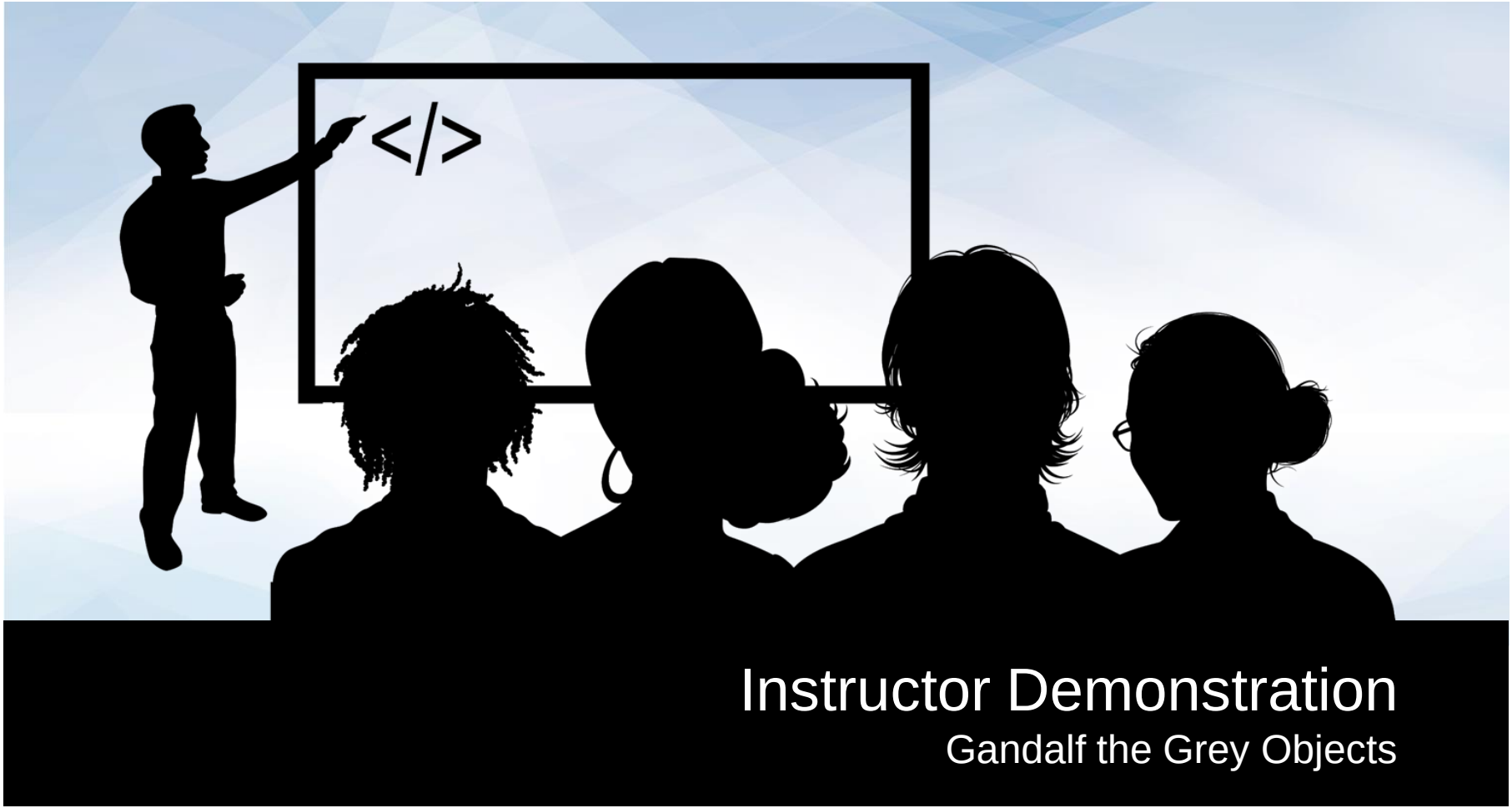
Joan of Arc (Bad Arrays)

Associated Data ==/== Arrays

Relating two separate arrays is not fun.

```
var joanOfArcInfoParts = ["Real Name", "Grew Up Where", "Known For", "Scars", "Symbolism"];


var joanOfArcInfoValues = ["Jehanne la Pucelle.", "Domremy, a village in northeastern France.",
  "Peasant girl, daughter of a farmer, who rose to become Commander of the French army.",
  "Took an arrow to the shoulder and a crossbow bolt to the thigh while trying to liberate Paris.",
  "Stands for French unity and nationalism."];
```



Instructor Demonstration

Gandalf the Grey Objects

Gandalf: The Object

Gandalf's **properties** and **values** are associated in object form, making it easy to recall specific data.

```
var gandalf = {
  "real name": "Gandalf",
  "age (est)": 11000,
  "race": "Maia",
  "haveRetirementPlan": true,
  "aliases": [
    "Greyhame",
    "Stormcrow",
    "Mithrandir",
    "Gandalf the Grey",
    "Gandalf the White"
  ]
}

// Object properties can be accessed with "bracket notation"
alert("My name is " + gandalf["real name"]);

// Or with "dot notation" if the property has no spaces
if (gandalf.haveRetirementPlan) {

  // Or with a variable that matches the name of the property
  var ageProperty = "age (est)";
  var years = gandalf[ageProperty];
  alert("My 401k has been gathering interest for " + years + " years!");
}
```

Objects Visualized

This is Gandalf. According to code, Gandalf is an **object**.

var gandalf	=	{

“real name”	:	“Gandalf”	,

“age (est)”	:	11000	,

“race”	:	“Maia”

}

Objects Visualized

These are Gandalf's **properties** (like descriptors).

var gandalf	=	{

“real name”	:	“Gandalf”	,
“age (est)”	:	11000	,
“race”	:	“Maia”	

}

Objects Visualized

These are the **values** of Gandalf's properties.

var gandalf	=	{

"real name"	:	"Gandalf"	,
"age (est)"	:	11000	,
"race"	:	"Maia"	

}

Objects Visualized

Thus: gandalf[“race”] = “Maia

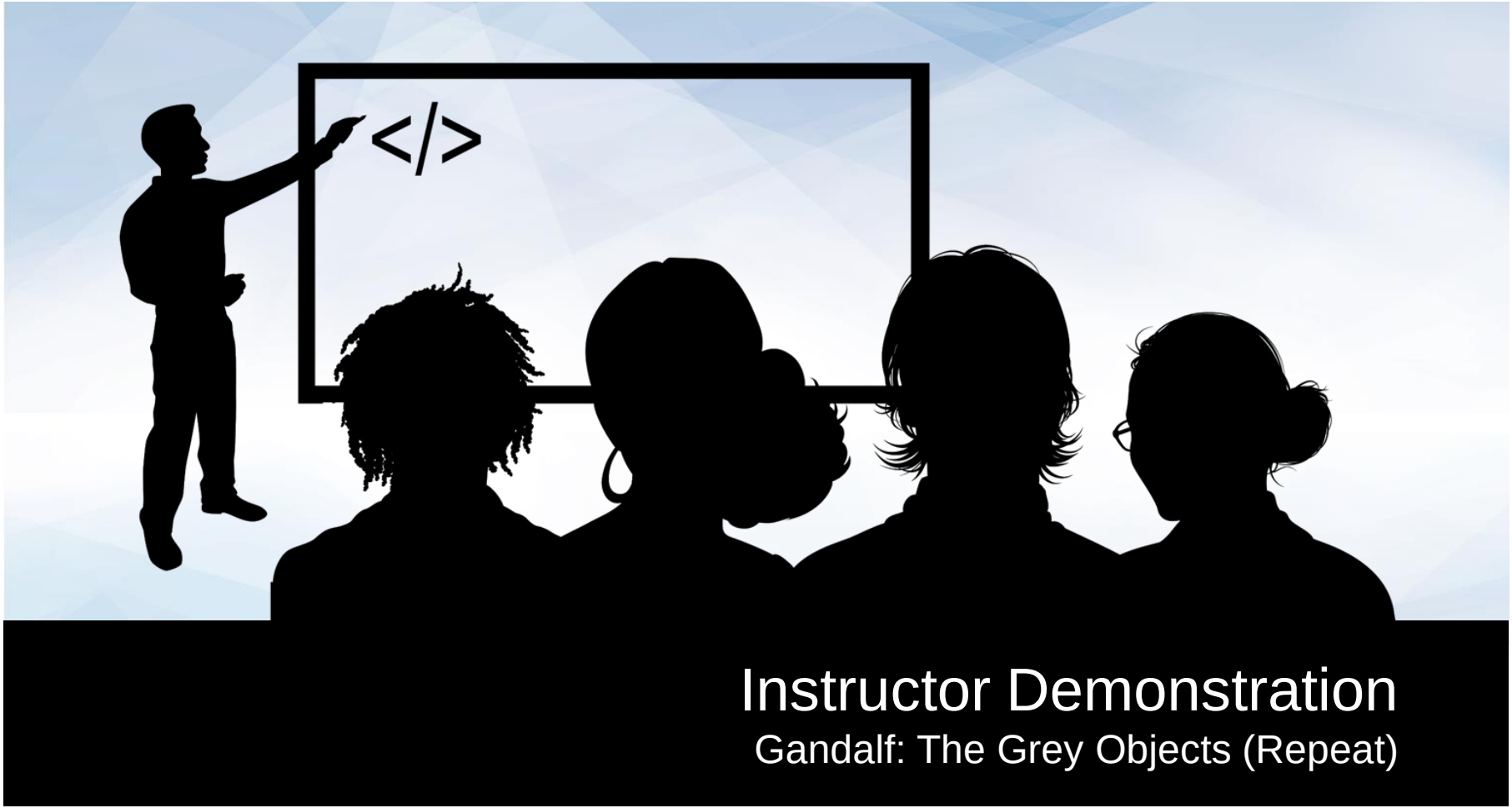
var gandalf	=	{

“real name”	:	“Gandalf”	,

“age (est)”	:	11000	,

“race”	:	“Maia”

}



Instructor Demonstration

Gandalf: The Grey Objects (Repeat)

Group Activity (2 people)**:** Basic Objects

Suggested Time:
15 minutes

Group Activity: Basic Objects

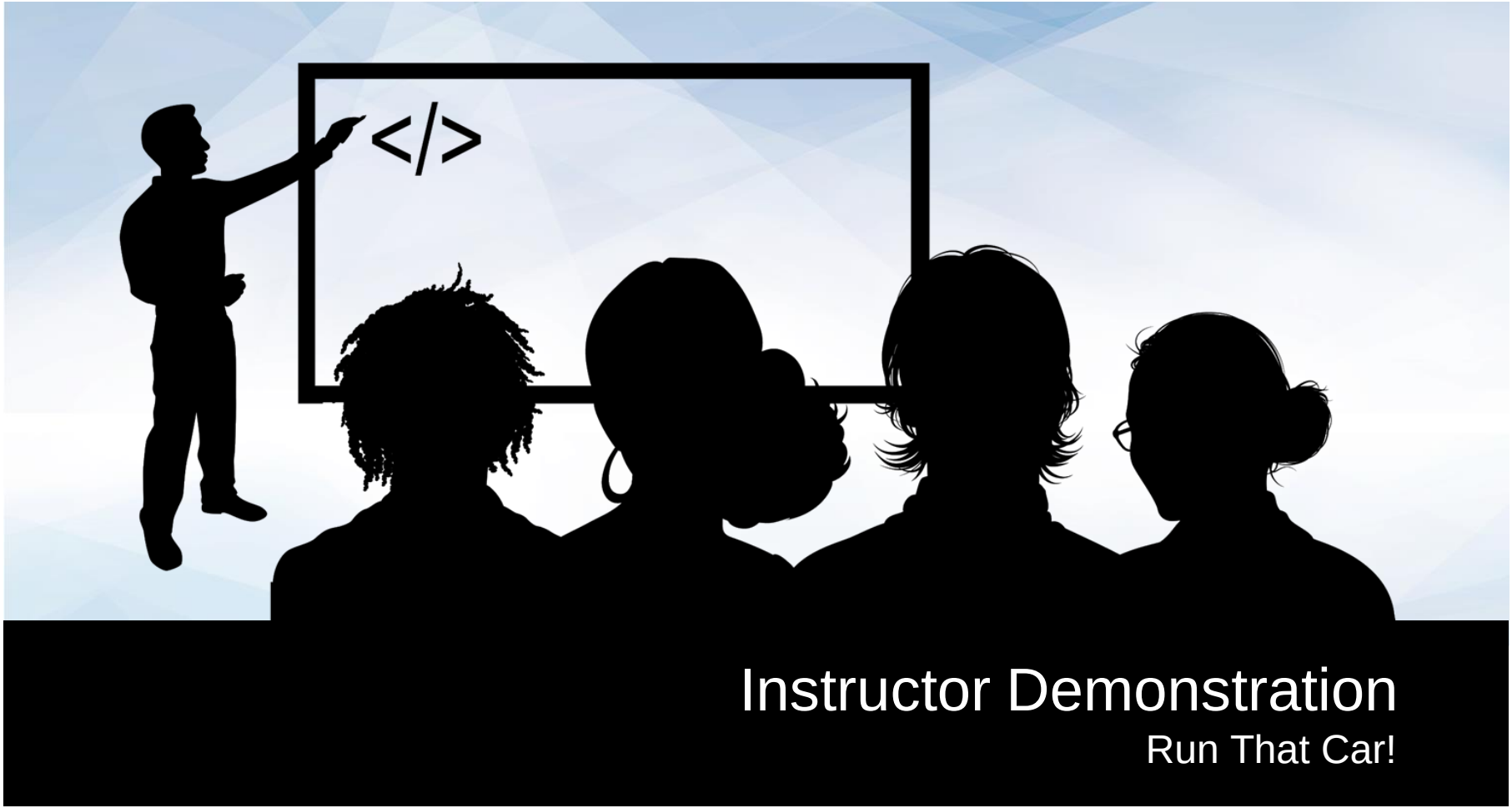
With a partner, spend the next few moments studying the code just slacked to you.

Then, write code below each comment to log the relevant information about the provided `car` object.

Bonus: If you finish early, create a brand new object of your own. Slack out a snippet of the code to the class when you are done. Be Creative!

Suggested Time: 15 minutes

Take a Break!



Instructor Demonstration

Run That Car!

Challenge: Run That Car!

Suggested Time:
15 minutes

Challenge: Run That Car!

Using the code from the previous activity as a starting point, create a complete application such that:

- Users can enter keyboard input (letters).
- Each of the car's methods are assigned to a key.
- When the user presses a key, it calls the appropriate function.
- These letters also trigger a global function called `rewriteStats()` that logs the car's make, model, color, mileage, and `isWorking` status to the console.

Suggested Time: 15 minutes

Questions?